# CS388G Problem Set #5

Nidhi Kadkol (nk9368)

April 2019

## 1

Our objective is to minimize $\sum_{e \in E} \alpha_e c(e)$ such that

$$\begin{aligned}
&\alpha_{(u,v)} + \delta_v - \delta_u \geq 0 \text{ for all } (u,v) \in E,\\
&\alpha_e \geq 0,\\
&0 \leq \delta_v \leq 1 \text{ for all } v \in V,\\
&\delta_s = 1,\\
&\delta_t = 0.
\end{aligned}$$

From the first constraint, we get that $\alpha_{(u,v)} \geq \delta_u - \delta_v$. Since the costs of the edges are fixed, we essentially want to minimize the value of each $\alpha_{(u,v)}$. Using this fact, we combine the first and second constraints to get

$$\alpha_{(u,v)} = \max(\delta_u - \delta_v, 0)$$

Let us suppose we have found an optimal value for the $\alpha$'s and $\delta$'s that minimizes the objective function. Now consider a vertex $u$ with $\delta_u$ being a part of the optimal solution, and $0 < \delta_u < 1$. We will argue that adding a small non-zero value $\epsilon$ to $\delta_u$ ($\epsilon$ is small enough such that adding it to $\delta_u$ will not give a value greater than $\delta_v$ for any $v$ in $V$) will still give us an optimal solution. There are 4 kinds of edges incident on $u$, we will look at them in cases.

**Case 1:** edge $e = (u,v), \delta_u \geq \delta_v$:

In this case, $\delta_u - \delta_v \geq 0$ and so $\alpha_{(u,v)} = \delta_u - \delta_v$. Adding $\epsilon$ to $\delta_u$ would make $\alpha'_{(u,v)} = \alpha(u,v) + \epsilon$, and hence $\alpha'_{(e)} c(e) = [\alpha_{(e)} + \epsilon]\, c(e)$. Let the sum of the costs of the edges in this set be $C_0$. So $\sum_{e:e=(u,v),\delta_u \geq \delta_v} \alpha'_{(e)} c(e) = \sum_{e:e=(u,v),\delta_u \geq \delta_v} \alpha_{(e)} c(e) + \epsilon\, C_0$.

Thus the change in the objective function from the edges in this set is $\epsilon\, C_0$.

**Case 2:** edge $e = (u,v), \delta_u < \delta_v$:

In this case, $\delta_u - \delta_v < 0$ and so $\alpha_{(u,v)} = 0$. If $\epsilon$ is negative, then adding it to $\delta_u$ would make $\delta_u - \delta_v$ more negative and $\alpha'_{(e)} = \alpha_{(e)} = 0$. If $\epsilon$ is positive, then adding it to $\delta_u$ would not make $\delta_u - \delta_v > 0$ as $\epsilon$ is small enough that adding it to $\delta_u$ will not give a value greater than $\delta_v$ for any $v$ in $V$. So $\alpha'_{(e)} = \alpha_{(e)} = 0$, and these edges do not contribute to changing the value of the objective function.

**Case 3:** edge $e = (v, u), \delta_u \leq \delta_v$: In this case, $\delta_v - \delta_u \leq 0$ and so $\alpha_{(v,u)} = \delta_v - \delta_u$. Adding $\epsilon$ to $\delta_u$ would make $\alpha'_{(v,u)} = \alpha(v, u) - \epsilon$, and hence $\alpha'_{(e)} c(e) = [\alpha_{(e)} - \epsilon]\, c(e)$. Let the sum of the costs of the edges in this set be $C_1$. So $\sum_{e:e=(v,u),\delta_u\leq\delta_v} \alpha'_{(e)} c(e) = \sum_{e:e=(v,u),\delta_u\leq\delta_v} \alpha_{(e)} c(e) - \epsilon\, C_1$.

Thus the change in the objective function from the edges in this set is $-\epsilon\, C_1$.

**Case 4:** edge $e = (v, u), \delta_u > \delta_v$: In this case, $\delta_v - \delta_u < 0$ and so $\alpha_{(v,u)} = 0$. If $\epsilon$ is positive, then adding it to $\delta_u$ would make $\delta_v - \delta_u$ more negative and $\alpha'_{(e)} = \alpha_{(e)} = 0$. If $\epsilon$ is negative, then adding it to $\delta_u$ would not make $\delta_v - \delta_u > 0$ as $\epsilon$ is small enough that adding it to $\delta_u$ will not give a value greater than $\delta_v$ for any $v$ in $V$. So $\alpha'_{(e)} = \alpha_{(e)} = 0$, and these edges do not contribute to changing the value of the objective function.

Thus, adding $\epsilon$ changes the value of our objective function by $\epsilon\,(C_0 - C_1)$. If $C_0 < C_1$, then a positive $\epsilon$ decreases the objective function, which contradicts the fact that it was originally optimal. Similarly, if $C_0 > C_1$, then a negative $\epsilon$ decreases the objective function, which contradicts the fact that it was originally optimal. Thus, $C_0 = C_1$, and we can add $\epsilon$ to $\delta_u$ and still have an optimal solution.

Thus, we can arrange all the vertices in increasing order of $\delta$, and can simultaneously add $\epsilon = 1 - \delta$ to the vertices with the highest $\delta$. This will make the $\delta$'s of these vertices equal to 1. Then we repeat the procedure for the set of next highest vertices, and so on, until we reach the vertices with the smallest $\delta > 0$. After adding $\epsilon = 1 - \delta$ to these vertices, we get a set of vertices where every vertex has a $\delta$ which is either 0 or 1, and the objective function with these assignments is optimal. Thus, the linear program admits a 0-1 optimal solution.

## 2

Let each edge be represented as $x_e$ or $x_{ij}$ from vertex $i$ to vertex $j$, and the cost associated with that edge be $c_e$. We formulate the min-cost perfect matching for a bipartite graph as the following Linear Program:

$$\min \sum_{e\in E} c_e x_e \quad \text{such that} \tag{1}$$

$$\sum_{j:(i,j)\in E} x_{ij} = 1 \quad \forall i \in X \tag{2}$$

$$\sum_{i:(i,j)\in E} x_{ij} = 1 \quad \forall j \in Y \tag{3}$$

$$x_e \in \{0, 1\} \qquad \forall e \in E \tag{4}$$

Solving this is NP-Hard, so we relax the last constraint to be $x_e \geq 0$. From constraints 2 and 3, it follows that $0 \leq x_e \leq 1 \forall e \in E$. This is a tractable problem. Now given a feasible instance of this problem, let us suppose we solve it and get a set of values for $x_e$ which optimize the objective function. Some of the $x_e$'s will be 0, some 1, and some will be between 0 and 1.

We construct a new graph $G'$ by discarding the edges which have $x_e$ equal to 0. This new graph is a bipartite graph, as we have taken a subset of the edges from the original bipartite graph and have not added any edges. Now, for each vertex in $X$, there has to be at least one edge incident on it, as otherwise constraint 2 will not be satisfied. Thus, for any subset of $X$ we have that the number of vertices adjacent to some member in the subset of $X$ is greater than or equal to the number of vertices in that subset of $X$. Thus by Hall's theorem, there exists a perfect matching $M$ in $G'$.

Let $a$ be the smallest value of $x_e$ in $G'$, and let the fractional matching in $G'$ by the $x_e$'s be denoted by $F$. Then from constraints 2 and 3, $F - aM$ is a set of edges in $G'$ such that the sum of the values of the edges incident on any vertex is $1 - a$. Also, at least one edge in $F - aM$ will have $x_e = 0$. We discard the edges which have $x_e = 0$ and also multiply the value of each edge with $1/(1-a)$. Then, this updated graph is also a feasible instance of the problem as now constraints 2 and 3 are satisfied (constraint 4 is satisfied both before and after multiplying by $1/(1-a)$). Thus, once again we can reason by Hall's theorem that a perfect matching exists in this updated graph. Hence, we can iteratively update the graph by subtracting the corresponding $aM$ from it. This update will terminate when we reach an empty graph, which is guaranteed as in every iteration at least one edge is dropped. Thus, we have been able to express our fractional matching as a convex sum of perfect matchings. Note that it has to be a convex sum and not merely a linear sum for constraint 2 and 3 to be satisfied.

Now we have that our fractional matching is a weighted average of perfect matchings, where the weights add up to 1. This means that the optimal objective (equation 1) for the relaxed LP is a weighted average of the objectives for the Integral LPs (whose feasible solutions are the corresponding perfect matchings). This means that the objectives for the perfect matching in the weighted average are all equal. If suppose they weren't equal then the value of the relaxed LP's objective is between the minimum and the maximum objectives of the perfect matchings in the weighted average. This is a contradiction, as the objective for the relaxed LP is optimal, and now we have found a matching corresponding to a lower objective. Thus, the objectives for the perfect matching in the weighted average are all equal and the objective for the relaxed LP is equal to these objectives. The perfect matchings corresponding to these objectives are all min cost perfect matchings. If suppose they weren't min cost then that means there exists a perfect matching with a lower objective than that of the relaxed LP, which is a contradiction. So these min cost perfect matchings are 0-1 optimal solutions to the feasible instance $I$ of the Linear Program. Thus, $I$ admits a 0-1 optimal solution.

## 3

We will show that the restricted version of 3-SAT belongs to P by reducing an instance of this problem to an instance of a $d$-regular bipartite graph matching problem.

Given a balanced 3-CNF formula $f$, we construct a graph which has each variable in $f$ as a vertex, as well as each clause in $f$ as a vertex. We draw an edge from a variable vertex to a clause vertex if that variable (whether negated or not) appears in that clause. Constructing this graph can be done in polynomial time as it requires one pass through the formula. Since each variable appears in exactly 3 clauses, each variable vertex will have exactly 3 edges incident on it which go to 3 different clause vertices. Also, since the variables in a clause are required to be distinct, each clause vertex will have exactly 3 edges incident on it which come from 3 different variable vertices. This means that the degree of every vertex in this graph is exactly 3. Also, there are no edges between variable vertices or between clause vertices. Hence, this is a 3-regular bipartite graph.

We can say that $f$ is satisfiable if there exists a truth assignment such that at least one literal in every clause is true. This is equivalent to finding a maximum matching in the graph we have constructed, as it will give us a mapping from each variable to a clause that is in present in. By assigning true to the literal corresponding to the variable in the clause it is matched to, its clause will be satisfied. So we can satisfy all the clauses if a matching that matches every clause vertex exists. Since the number of clause vertices is equal to the number of variable vertices since it is a $d$-regular bipartite graph, it is equivalent to finding a maximum matching. If such a matching does not exist, then the formula $f$ is unsatisfiable.

Since restricted 3-SAT $\leq_P$ maximum bipartite matching problem, we know that solving the 3-SAT problem is no harder than solving the matching problem. Since we can solve the matching problem in polynomial time, therefore we can solve the restricted 3-SAT problem in polynomial time as well.

From the result of exercise 26.3-5, we know that a matching of cardinality equal to the number of variable vertices exists. This means that the balanced 3-CNF is always satisfiable, and hence we will always return true. Thus solving this problem can be done in polynomial time and hence the restricted 3-SAT problem belongs to P.

## 4

Nice (2,3) SAT can be verified in polynomial time, and hence it is NP. To show that nice (2,3) SAT is NP-complete, we reduce 3-SAT to nice (2,3) SAT. Since 3-SAT is NP-complete and 3-SAT $\leq_P$ nice (2,3) SAT, therefore nice (2,3) SAT is NP-complete. We now show how to carry out the reduction.

Consider a 3-CNF formula $F$. Suppose a variable $x$ appears in 2 or more clauses in $F$ as $x$. Let the number of clauses in which $x$ appears be $k$ where $k \geq 2$. We shall replace the $k$ occurrences of $x$ with new variables $y_1, y_2, \ldots, y_k$. We shall also extend $F$ by AND-ing it with $G$:

$$G = ((\neg x \vee y_1) \wedge (\neg y_1 \vee y_2) \wedge \cdots \wedge (\neg y_k \vee x))$$

Consider the case when

$$y_1 = y_2 = \cdots = y_k = x \tag{5}$$

Then $G$ becomes

$$((\neg x \vee x) \wedge (\neg x \vee x) \wedge \cdots \wedge (\neg x \vee x))$$

which is always true.

Now suppose (5) is not true, that is, there is some $y_i \neq x$. So $y_i = \neg x$ and $\neg y_i = \neg\neg x = x$. So then $G$ is

$$\begin{aligned}
&((\neg x \vee y_1) \wedge (\neg y_1 \vee y_2) \wedge \cdots \wedge (\neg y_{i-1} \vee y_i) \wedge (\neg y_i \vee y_{i+1}) \wedge \cdots \wedge (\neg y_k \vee x))\\
=&((\neg x \vee x) \wedge (\neg x \vee x) \wedge \cdots \wedge (\neg x \vee \neg x) \wedge (x \vee x) \wedge \cdots \wedge (\neg x \vee x))\\
=&(t \wedge t \wedge \cdots \wedge \neg x \wedge x \wedge \cdots \wedge t) \ \text{ (since } (\neg x \vee x) \text{ evaluates to true for any assignment of } x)\\
=&(\neg x \wedge x) \ \text{ (since } x \wedge t = x \text{ for any formula } x)
\end{aligned}$$

This is always false. Even if there is more than one $y_i \neq x$, by the same line of reasoning as above, $G$ will evaluate to false.

Now we will show that $F$ is satisfiable if and only if $F \wedge G$ is satisfiable.

Suppose $F$ is satisfiable. Then there exists an assignment to the variables in $F$ for which $F$ evaluates to true. We can set $y_1 = y_2 = \cdots = y_k = x$, where $x$ gets the truth value which it does in the assignment that satisfies $F$. Thus, $F$ and $G$ are true, which means $F \wedge G$ is true. Thus, $F \wedge G$ is satisfiable.

Suppose $F$ is unsatisfiable. Then every assignment of truth values to the variables in $F$ will result in $F$ being false. Thus, every assignment of variables will result in $F \wedge G$ being false. Thus, there is no assignment which satisfies $F \wedge G$ and hence $F \wedge G$ is unsatisfiable.

Thus we can modify $F$ to a nice (2,3) CNF $F \wedge G$ which is satisfiable iff $F$ is satisfiable. Furthermore, $F \wedge G$ contains exactly one occurrence of $x$, and we have increased the number of occurrences of $\neg x$ by one. We can do this for each of the variables $x$ that appear more than twice as $x$ in $F$ to get a final nice (2,3) CNF formula $F'$.

Now consider the nice (2,3) CNF $F'$. Suppose a variable $x$ appears in more than 2 clauses in $F'$ as $\neg x$. Let the number of clauses in which $x$ appears be $j$ where $j \geq 2$. We shall replace the $j$ occurrences of $x$ with new variables $z_1, z_2, \ldots, z_j$. We shall also extend $F'$ by AND-ing it with $G'$:

$$G' = ((x \vee z_1) \wedge (\neg z_1 \vee z_2) \wedge \cdots \wedge (\neg z_j \vee \neg x))$$

Consider the case when

$$z_1 = z_2 = \cdots = z_j = \neg x \tag{6}$$

Then $G'$ becomes

$$((x \lor \neg x) \land (x \lor \neg x) \land \cdots \land (x \lor \neg x))$$

which is always true.

Now suppose (6) is not true, that is, there is some $z_i \neq \neg x$. So $z_i = x$ and $\neg z_i = \neg x$. So then $G'$ is

$$\begin{aligned}
&((x \lor z_1) \land (\neg z_1 \lor z_2) \land \cdots \land (\neg z_{i-1} \lor z_i) \land (\neg z_i \lor z_{i+1}) \land \cdots \land (\neg z_j \lor \neg x))\\
=&((x \lor \neg x) \land (x \lor \neg x) \land \cdots \land (x \lor x) \land (\neg x \lor \neg x) \land \cdots \land (x \lor \neg x))\\
=&(t \land t \land \cdots \land x \land \neg x \land \cdots \land t) \text{ (since } (x \lor \neg x) \text{ evaluates to true for any assignment of } x)\\
=&(x \land \neg x) \text{ (since } x \land t = x \text{ for any formula } x)
\end{aligned}$$

This is always false. Even if there is more than one $x_i \neq \neg x$, by the same line of reasoning as above, $G'$ will evaluate to false.

Now we will show that $F'$ is satisfiable if and only if $F' \land G'$ is satisfiable.

Suppose $F'$ is satisfiable. Then there exists an assignment to the variables in $F'$ for which $F'$ evaluates to true. We can set $z_1 = z_2 = \cdots = z_j = \neg x$, where $x$ gets the truth value which it does in the assignment that satisfies $F'$. Thus, $F'$ and $G'$ are true, which means $F' \land G'$ is true. Thus, $F' \land G'$ is satisfiable.

Suppose $F'$ is unsatisfiable. Then every assignment of truth values to the variables in $F'$ will result in $F'$ being false. Thus, every assignment of variables will result in $F' \land G'$ being false. Thus, there is no assignment which satisfies $F' \land G'$ and hence $F' \land G'$ is unsatisfiable.

Thus we can modify $F'$ to a nice (2,3) CNF $F' \land G'$ which is satisfiable iff $F'$ is satisfiable. Furthermore, $F' \land G'$ contains exactly one occurrence of $\neg x$, and we have increased the number of occurrences of $x$ by one. Since from the previous step, the number of occurrences of $x$ is at most 1, thus after this step the number of occurrences of $x$ is at most 2. Thus, no literal appears in more than 2 clauses. We can do this for each of the variables $x$ that appear two or more times as $\neg x$ in $F'$ to get a final nice (2,3) CNF formula $F''$.

Thus, we have reduced a 3-CNF formula $F$ to a nice (2,3) CNF formula $F''$ such that $F$ is satisfiable iff $F''$ is satisfiable. Thus, any instance of the 3-SAT problem can be reduced to an instance of the nice (2,3)-SAT problem and hence the nice (2,3)-SAT problem is NP-complete.